Computational Intelligence
AN INTERNATIONAL JOURNAL

Computational Intelligence, Volume 32, Number 3, 2016

DOCUMENT CLUSTERING WITH DUAL SUPERVISION THROUGH FEATURE REWEIGHTING*

YEMING HU,[1] EVANGELOS E. MILIOS,[1] AND JAMES BLUSTEIN[2]

[1]*Faculty of Computer Science, Dalhousie University, Nova Scotia, Canada*

[2]*Faculty of Computer Science and School of Management, Dalhousie University, Nova Scotia, Canada*

Traditional semi-supervised clustering uses only limited user supervision in the form of instance seeds for clusters and pairwise instance constraints to aid unsupervised clustering. However, user supervision can also be provided in alternative forms for document clustering, such as labeling a feature by indicating whether it discriminates among clusters. This article thus fills this void by enhancing traditional semi-supervised clustering with feature supervision, which asks the user to label discriminating features during defining (labeling) the instance seeds or pairwise instance constraints. Various types of semi-supervised clustering algorithms were explored with feature supervision. Our experimental results on several real-world data sets demonstrate that augmenting the instance-level supervision with feature-level supervision can significantly improve document clustering performance.

Received 3 September 2012; Revised 21 May 2014; Accepted 19 December 2014

Key words: user supervision, feature supervision, feature reweighting, text cloud.

1. INTRODUCTION

Traditional document clustering is an unsupervised categorization of a given document collection into clusters so that documents within the same cluster are more topically similar than those in different clusters. However, given a document collection, different users may want it organized in their own point of view instead of a universal one. Consider a collection of news articles about international sports. One user may like to organize the collection by country, while another may want it organized by sport, in which unsupervised clustering is incapable. This is addressed by incorporating user supervision into the clustering process. In this article, we use two types of user supervision, i.e., document supervision and feature supervision for document clustering. *Document supervision* involves labeling (defining) documents, i.e., assigning a document to a cluster (defining a document seed) or specifying a pairwise constraint "must-link" or "cannot-link" (Wagstaff et al. 2001) between two documents (defining a document constraint). *Feature supervision* involves labeling features, i.e., indicating whether a feature discriminates clusters. We say a feature is accepted if it is labeled as discriminating. Note that accepted features are not assigned to a cluster but known for their usefulness in clustering.

Traditional semi-supervised clustering, which uses both labeled and unlabeled instances, has shown its usefulness in generating clusters matching user expectations. User supervision usually takes the form of document supervision. In these methods, the user defines instance seeds for initializing clusters or provides an instance constraint by indicating whether the two instances involved should be placed into the same cluster or not.

Address correspondence to Yeming Hu, Faculty of Computer Science, Dalhousie University, 6050 University Avenue Halifax, Nova Scotia, Canada; e-mail: yeming@cs.dal.ca

*A short version of this work was published as Hu, Milios, and Blustein (2012). Compared with the short version, this version includes the following: (1) more details on the clustering methods with labeled documents on which we develop our framework; (2) the models for dual supervision, i.e., document supervision and feature supervision; and (3) more results on evaluating the proposed framework.

(a) Text Cloud of Document A about Canadian Basketball

(b) Text Cloud of Document B about Canadian Hockey

FIGURE 1. Text clouds of two documents. (a) Text cloud of document A about Canadian basketball. (b) Text cloud of document B about Canadian hockey.

However, the user can also provide alternative forms of user supervision such as feature supervision involving labeling features for document clustering. Because this article focuses on document clustering, we may use *instance* and *document*, and *feature* and *word* interchangeably. Labeling documents and words can be performed at the same time with little additional effort for labeling words, if an appropriate document visualization is used, such as text clouds (Lamantia 2007). While the user assigns a document to a cluster or specifies a pairwise constraint based on the document's text cloud, the words appearing in the text cloud can also be labeled by being clicked or highlighted.

Example 1. Documents A and B in Figure 1 can be specified as a must-link when clustered by country but a cannot-link when clustered by sport. Correspondingly, the user would accept the words "Canada," "Canadian," and "Spain" in the first case but "basketball," "points," "hockey," and "rychel" (last name of a hockey player) in the latter case.

Different accepted words reflect different organizations, and the user forms his or her point of view based on the perception of the words in the text clouds. It has been argued that document supervision and feature supervision are complementary rather than completely redundant, and their joint use has been called *dual supervision* (Attenberg, Melville, and Provost 2010).

In this article, we assume that the user defines a document seed or establishes a pairwise constraint by reading a fraction of the documents' contents. At the same time, the

user can label a word by indicating (e.g., highlighting) whether it discriminates among clusters. The text cloud could be used to visualize the fraction of the content and augment the labeling process. Because the accepted features are not associated with specific clusters, we incorporate them into the semi-supervised clustering through feature reweighting. Despite its simplicity, our proposed method is proven to be quite robust and effective under different experimental settings. We enhance semi-supervised clustering algorithms in different categories mentioned in Section 2. We also compare those algorithms using only document seeds or document constraints with our proposed method using only accepted features. Finally, we performed experiments by allowing the user to make errors in accepting features and to only read a fraction of a document content and by allowing for various numbers of documents to be assigned to each cluster.

The rest of this article is organized as follows. Related work on semi-supervised clustering and feature supervision is discussed in Section 2. In Section 3, we introduce some background knowledge and describe the unsupervised and semi-supervised clustering algorithms we enhance with feature supervision. In Section 4, we present the methodology for incorporating the feature supervision. The details of the experimental results on several real-world text data sets are presented and discussed in Section 5. We conclude this article and discuss future work in Section 6.

2. RELATED WORK

Existing semi-supervised clustering techniques, employing user supervision in the form of instance-level constraints, are generally grouped into four categories. First, constraints are used to modify the loss function (Basu, Bilenko, and Mooney 2004; Ji and Xu 2006; Yoshida 2012). Second, cluster seeds derived from the constraints initialize the cluster centers (Basu, Banerjee, and Mooney 2002). Third, constraints are employed to learn adaptive distance metrics using metric learning techniques (Cheng, Hua, and Vu 2008). Finally, the original high-dimensional feature space can be projected into low-dimensional feature subspaces guided by constraints (Tang et al. 2007). In this article, we enhance the first three methods with user-accepted features obtained from feature supervision.

Liu et al. (2004) propose to ask the user to assign features with class labels and use the set of features accepted for each class to find a set of documents for training classifiers. Druck, Mann, and McCallum (2008) use accepted features with class labels to constrain the probabilistic model estimation on unlabeled instances instead of creating pseudo-instances as carried out in other approaches. Raghavan, Madani, and Jones (2005) make use of feature feedback in the active learning with support vector machine by up-weighting the accepted features. All those methods ask the user to assign class labels to features and require accepted features for each class. In our article, we do not ask the user to label features with cluster labels. In fact, the user does not even give the cluster label for a feature but just indicates whether it is useful for clustering. We also assume that the user has the document content as context to label words instead of having a stand-alone ranked list of features from which to label features. In addition, the accepted features are used to modify document representations when cluster labels of the features are not given. Huang and Mitchell (2006) propose a generative probabilistic framework to incorporate various types of user feedback including feedback on features. In their work, the user needs to assign a feature to an intermediate cluster, while we only ask the user to indicate whether a feature is good or not for clustering. Hu, Milios, and Blustein (2011) propose an interactive framework for feature selection for document clustering, in which the user only indicates whether a feature is suitable for clustering. However, their work asks the user to label features from a stand-alone ranked list of features. More importantly, they did not explore the usefulness

of integrating document seeds (or document constraints) and features together or compare feature supervision and document supervision for clustering.

3. BACKGROUND

In this section, we introduce pairwise document constraints and present three semi-supervised clustering algorithms enhanced, each of which is from a different category in Section 2. COP K-means is a constraint-based method, while seeded K-means uses documents as cluster seeds for document clustering. Constrained K-means uses documents for both cluster seeds and constraints derived from document seeds. Xing K-means is a distance metric learning method.

3.1. Pairwise Document Constraints

Two types of pairwise constraints are used for traditional semi-supervised clustering:

- *Must-link* constraints specify that two documents have to be placed in the same cluster.
- *Cannot-link* constraints specify that two documents cannot be placed in the same cluster.

Consequently, there are usually two sets defined: $\mathcal{M}$ is the set of must-link constraints, and $\mathcal{C}$ is the set of cannot-link constraints. Both must-link and cannot-link constraints are symmetric. Ideally, the must-link constraints are transitive, and transitive closures can be derived.

3.2. K-Means

K-means is a clustering algorithm based on iterative assignments of data points to clusters and partitions a data set into K clusters so that the average squared distance between the data points and the closest cluster centers is locally minimized. For a data set with data points $\mathcal{X} = \{x_1, x_2, \ldots, x_N\}, x_i \in \mathcal{R}^d$, K-means algorithm generates K clusters $\{\mathcal{X}_l\}_{l=1}^K$ of $\mathcal{X}$ such that the objective function

$$J = \sum_{l=1}^{K} \sum_{x_i \in \mathcal{X}_l} ||x_i - \mu_l||^2 \tag{1}$$

is locally minimized and $\{\mu_1, \mu_2, \ldots, \mu_K\}$ represent the centers of the K clusters.

3.3. COP K-Means

COP K-means (Wagstaff et al. 2001) is a constraint-based method, where user supervision is provided in the form of must-link and cannot-link constraints. During the clustering process, all the constraints should be satisfied. Otherwise, COP K-means fails, in which no clusters are produced. COP K-means is presented in Algorithm 1.

3.4. Seeded K-Means

Given a data set $\mathcal{X}$, K-means can partition it into K clusters $\{\mathcal{X}_l\}_{l=1}^K$. K-means is usually initialized with randomly selected cluster centers. It was observed that seeded K-means (Basu et al. 2002), with cluster centers initialized with centroids derived from small sets of instances, could improve clustering performance significantly. To this end, we define the seed set $\mathcal{S} \subseteq \mathcal{X}$ to be the subset of data points as follows: for each $x_i \in \mathcal{S}$, the user

Algorithm 1 COP K-means

Input: Set of data points $\mathcal{X}$, must-link set $\mathcal{M}$, cannot-link set $\mathcal{C}$
Output: K clusters $\{\mathcal{X}_l\}_{l=1}^{K}$
Method:

1: Randomly initialize the cluster centers with $\{\mu_l^0\}_{l=1}^{K}$
2: **repeat**
3: Based on $\{\mu_l^t\}$, assign each data point x_i to the closest cluster $\mathcal{X}_l^{t+1}$ for which VIOLATE-CONSTRAINTS($x_i, \mathcal{X}_l, \mathcal{M}, \mathcal{C}$) (Algorithm 2) returns false. If no such cluster exists, COP K-means fails and returns {}. At the end, obtain $\{\mathcal{X}_l^{t+1}\}_{l=1}^{K}$.
4: Update cluster centers: $u_l^{(t+1)} \leftarrow \frac{1}{|\mathcal{X}_l^{(t)}|} \sum_{x \in \mathcal{X}_l^{(t)}} x$
5: $t \leftarrow t + 1$
6: **until** No data point assignments change or the maximum number of iterations is reached, where the maximum number of iterations is user defined based on heuristics.

Algorithm 2 VIOLATE-CONSTRAINTS

Input: Data points x_i, cluster $\mathcal{X}_l$ must-link set $\mathcal{M}$, and cannot-link set $\mathcal{C}$
Output: TRUE/FALSE
Method:

1: VIOLATED $\leftarrow$ FALSE
2: **for all** $(x_i, y) \in \mathcal{M}$ **do**
3: **if** $y \notin \mathcal{X}_l$ and y is already re-assigned in the current iteration; i.e., y is processed before x_i **then**
4: VIOLATED $\leftarrow$ TRUE;
5: break;
6: **end if**
7: **end for**
8: **for all** $(x_i, y) \in \mathcal{C}$ **do**
9: **if** $y \in \mathcal{X}_l$ and y is already re-assigned in the current iteration, i.e., y is processed before x_i **then**
10: VIOLATED $\leftarrow$ TRUE;
11: break;
12: **end if**
13: **end for**
14: return VIOLATED

provides the cluster $\mathcal{X}_l$ to which it belongs. We assume that there is at least one data point x_i for each cluster $\mathcal{X}_l$. Note that there is a K-disjoint partitioning $\{\mathcal{S}_l\}_{l=1}^{K}$ of the seed set $\mathcal{S}$ such that all $x_i \in \mathcal{S}_l$ belong to $\mathcal{X}_l$ according to the supervision. In seeded K-means (Basu et al. 2002), the seed set $\mathcal{S}$ is used to initialize the K-means algorithm. In this method, each cluster center μ_l is initialized by the centroid of $\mathcal{S}_l$ instead of a randomly picked centroid. Note that the seed set is only used in the initialization step and is not used in the remaining steps of K-means. Therefore, the seeds can change their cluster memberships during the subsequent clustering steps. In constrained K-means (Basu et al. 2002), the seed set is also used to initialize the K-means algorithm. However, unlike seeded K-means, the memberships of the seeds are not re-computed and kept unchanged in the subsequent clustering steps. Compared with seeded K-means, constrained K-means is more appropriate when there are no or very few noisy seeds. In fact, constrained K-means can be thought of as a combination of COP

K-means and seeded K-means. In this article, we assume the seed set without noise. Seeded K-means and constrained K-means are described in Algorithm 3.

Algorithm 3 Seeded K-means and Constrained K-means

Input: Set of data points $\mathcal{X}$, the seed set $\mathcal{S} = \cup_{l=1}^{K} \mathcal{S}_l$
Output: K clusters $\{\mathcal{X}_l\}_{l=1}^{K}$
Method:

initialize each cluster center μ_l using the seed subset $\mathcal{S}_l$: $\mu_l^0 \leftarrow \frac{1}{|\mathcal{S}_l|} \sum_{x \in \mathcal{S}_l} x$
$t \leftarrow 0$
repeat
  for all $x_i \in \mathcal{X}$ **do**
    if Constrained K-means and $x_i \in \mathcal{S}$ **then**
      Assign x_i to l where $x_i \in \mathcal{S}_l$
    else
      Assign x_i to the closest cluster $\mathcal{X}_l^{(t+1)}$ based on $\{\mu_l^t\}$ and obtain $\{\mathcal{X}_l^{(t+1)}\}_{l=1}^{K}$
    end if
  end for
  Update cluster centers: $u_l^{(t+1)} \leftarrow \frac{1}{\left|\mathcal{X}_l^{(t)}\right|} \sum_{x \in \mathcal{X}_l^{(t)}} x$
  $t \leftarrow t + 1$
until No data point assignments change or the maximum number of iterations is reached, where the maximum number of iterations is user defined based on heuristics.

3.5. Xing K-Means

Many clustering algorithms, including K-means, critically rely on a good metric for the input data. A better metric may be learned from the document pairwise constraints. Xing et al. (2003) provide a method to learn a generalized Euclidean distance metric based on the pairwise constraints. Because the learned Euclidean distance metric can be used as a component of K-means, we call it Xing K-means. Assuming the data set $\mathcal{X}$, must-link set $\mathcal{M}$, and cannot-link set $\mathcal{C}$, the distance metric $dst(x, y)$ between data points x and y can be written in the form of

$$dst(x, y) = dst_A(x, y) = \|x - y\|_A = \sqrt{(x - y)^T A (x - y)}. \tag{2}$$

The metric learning algorithm tries to learn a positive semi-definite A so that the following optimization problem is satisfied:

$$\min_{A} \sum_{(x_i, x_j) \in M} \|x_i - x_j\|_A^2 \tag{3}$$

$$\text{s.t.} \sum_{(x_i, x_j) \in C} \|x_i - x_j\|_A \geq 1. \tag{4}$$

$$A \succeq 0. \tag{5}$$

The choice of the constant 1 in the right-hand side of equation (4) is arbitrary, and it can be any positive constant c so long as A is replaced by $c^2 A$.

Because document vectors $\{x_i\}_{i=1}^{N}$ have very high dimensions, it is computationally prohibitive to estimate the full matrix A. Therefore, we only consider the case when the matrix A is diagonal. When A is a diagonal matrix, it can be represented as $A = diag(A_{11}, A_{22}, \ldots, A_{nn})$. An efficient algorithm for estimating A can be derived with the Newton–Raphson method by defining

$$\begin{aligned} g(A) &= g(A_{11}, A_{22}, \ldots, A_{nn}) \\ &= \sum_{x_i, x_j \in M} \|x_i - x_j\|_A^2 \\ &\quad - \log\left(\sum_{(x_i, x_j) \in C} \|x_i - x_j\|_A \right). \end{aligned} \tag{6}$$

It can be shown that minimizing g (subject to $A \succeq 0$) is equivalent, up to a multiplication of A by a positive constant to solve the optimization problem defined by equations (3)–(5) (Xing et al. 2003).

4. METHODOLOGY

In this section, we describe the details of the document oracle and feature oracle and present the model for labeling documents and features together. We propose a framework to incorporate feature supervision through feature reweighting into various traditional semi-supervised clustering algorithms introduced in Section 3. In the traditional semi-supervised clustering, only document seeds or constraints from document supervision are used to either initialize the clustering algorithms or guide the clustering process. In our framework, we introduce one more supervision dimension, namely, accepted features from feature supervision, into the traditional semi-supervised clustering algorithms.

4.1. Oracles

Designing the document oracle is straightforward because all the documents in our data sets have class labels. Therefore, the underlying document class labels can act as a document oracle (Basu et al. 2002; Basu et al. 2004; Ji and Xu 2006; Tang et al. 2007; Cheng et al. 2008; Attenberg et al. 2010). However, this is not the case for labeling features. Ideally, we should have a gold-standard feature set. To simulate how a human responds to queries for feature labels, we construct a feature oracle similarly to previous approaches (Druck et al. 2008; Attenberg et al. 2010). The χ^2 value of words with respect to the known labels in the document collection is computed, and all the words are ranked by their χ^2 values. Then, the top f words are taken as the feature oracle and will be regarded as useful for clustering when the user is queried about them. We define f as the capacity of the feature oracle. The larger f is, the more features this oracle can accept. However, because more noisy features may be included when f is large, we should select a proper f for our experiments. We investigate how the value of f affects clustering performance in our experiments. Because human users can make mistakes, we can also construct a noisy feature oracle. The bottom half of features in the list of features sorted by the χ^2 values with respect to the known labels are considered as noisy features. A noisy feature oracle can be constructed by replacing a certain number of features in the top f features by the same number of features in the bottom half of the list.

The construction of a feature oracle is presented in Algorithm 4. Note that our feature oracle is different from those previous feature oracles (Druck et al. 2008; Attenberg et al. 2010) in two aspects: (1) Our feature oracle only indicates whether a feature is useful for clustering instead of giving the feature a class/cluster label; and (2) Our feature oracle can be noisy by introducing $p_n f$ noise features (accepted by mistake) with a given probability $p_n > 0$.

Algorithm 4 Construction of a Feature Oracle

Data Input: Set of unordered features $\mathcal{F}$, training set $\mathcal{CL}$—documents and their class labels in the data set
Parameter Input: Noise level p_n, the percentage of noise features the feature oracle will mislabel as "accept," Feature Oracle Capacity f—the number of features the oracle labels as "accept," $f \ll |\mathcal{F}|$
Output: List of ordered features $\mathcal{L}$—the list of features the feature oracle labels as "accept"
Method:

1: Compute χ^2 values of all features in $\mathcal{F}$ based on $\mathcal{CL}$
2: Sort all features in $\mathcal{F}$ according to the computed χ^2 values and obtain ordered list $\mathcal{T}$ of the same size as $\mathcal{F}$
3: **for** $i = 1$ to f **do**
4: Flip a coin with the probability p_n getting the tail and obtain the outcome O
5: **if** O is tail **then**
6: Randomly pick a feature from the bottom half of $\mathcal{T}$, which is considered to be a noisy feature
7: Swap i^{th} feature with the picked noisy feature in $\mathcal{T}$
8: **end if**
9: **end for**
10: Generate $\mathcal{L}$ by taking the top f features of $\mathcal{T}$

4.2. Model for Document Supervision

Our purpose in this article is to demonstrate that enhancing traditional semi-supervised document clustering with feature supervision can improve the clustering performance. Therefore, with respect to document supervision in our framework, we adopt the same supervision methods using the document oracle as in the traditional semi-supervised clustering algorithms. In a seeded K-means-based framework, the seeds are randomly sampled from the documents belonging to the corresponding class according to the size of the seed set for each cluster. In a COP K-means-based framework, the must-link and cannot-link constraints are derived from the sampled seeds for seeded K-means; i.e., the seeds for the same clusters form the must-link constraints, while the seeds for the different clusters form the cannot-link constraints. In a distance metric learning-based framework, we randomly sample the designated number of constraints.

4.3. Model for Feature Supervision

We assume that the document (or document constraint) labeling and feature labeling happen simultaneously; i.e., the user can label words as useful for clustering, while he or she is defining a document seed or a pairwise document constraint, e.g., through highlighting keywords for a document. With this labeling model, a feature is accepted once the user recognizes it as useful for clustering while reading a document. The advantage of this labeling

model is saving user effort, because the user does not need to label the features separately. The disadvantage is that the user does not need to read the whole document content to establish a document constraint so that some useful features might be ignored. In our labeling model, we first assume the user will read the whole document content to define a document constraint. Then, we consider the user reading only the first fraction $p\%$ of content to define a document seed or document constraint.

For example, a document d can be considered as a list of words in the order in which the words occur in the document, i.e., $< w_1, w_2, \ldots, w_{|d|} >$, where $|d|$ is the length of the document in terms of the number of words. Note that w_i might be the same as w_j where i is not equal to j, $1 \leq i \leq |d|$ and $1 \leq j \leq |d|$. To define a document seed or a document constraint, we assume that the user needs to read at least a fraction of the document content p_c, i.e., $< w_s, w_{s+1}, \ldots, w_e >$, where $1 \leq s \leq |d|$, $s \leq e \leq |d|$, and $e - s + 1 = \lceil p_c \cdot |d| \rceil$. When $s = 1$, the user reads a document from the beginning. While reading a document, the user is assumed to be able to label words she or he encounters. The accepted words are included in the accepted feature set $\mathcal{W}^{\mathcal{L}}$. The fraction of document content could be displayed as a text cloud, and the user could accept words by highlighting them on the text clouds. The user accepts a feature if it is a good description of the topic of a cluster and discriminates the cluster from others. Note that the user does not need to associate a feature with a specific cluster.

Definition 1. Assuming accepted feature set $\mathcal{W}^{\mathcal{L}} = \{w | M(w) = \textit{labeled}\}$, where M is the function to produce the label of a feature:

$$M(w) = \begin{cases} \textit{labeled} & \text{if } w \text{ is confirmed as useful for clustering} \\ \textit{unlabeled} & \text{otherwise, i.e., } w \text{ is not presented or not confirmed as useful.} \end{cases} \tag{7}$$

4.4. Feature Reweighting

Because the feature reweighting employed by Hu et al. (2011) is simple and effective, we use it on the accepted features for semi-supervised clustering. Although different semi-supervised clustering algorithms may have their own method of integrating the feature reweighting, we only have one underlying algorithm K-means in this article.

Feature reweighting for K-means is achieved through reweighting the *TFIDF* (term frequency-inverse document frequency) values of features. More specifically, it is performed as follows: the *TFIDF* values of accepted features in $\mathcal{W}^{\mathcal{L}}$ are multiplied by a given weight g (>1):

$$R_w^{d_i}(\textit{tfidf}) = \begin{cases} O_w^{d_i}(\textit{tfidf}) \times g & \text{if } w \in \mathcal{W}^{\mathcal{L}} \\ O_w^{d_i}(\textit{tfidf}) & \text{otherwise} \end{cases} \tag{8}$$

where $O_w^{d_i}(\textit{tfidf})$ and $R_w^{d_i}(\textit{tfidf})$ are the original and reweighted *tfidf* values of feature w in document d_i, respectively. After being reweighted, the vector of *TFIDF* values is normalized. Because Xing K-means learns the feature weights based on the pairwise constraints, we use another heuristic to incorporate the accepted features. We perform Euclidean distance metric learning and obtain the feature weights. The most useful features based on the document constraints are assigned the highest weight by the metric learning algorithm. Because accepted features are regarded as useful for clustering by the user, it is reasonable to assign the highest weight to all accepted features $\in \mathcal{W}^{\mathcal{L}}$.

4.5. Semi-supervised Clustering with Feature Supervision

The procedure of semi-supervised clustering with feature supervision is presented in Algorithm 5. Because traditional semi-supervised clustering methods employ user supervision in the form of pairwise constraints or cluster seeds, adding feature supervision to semi-supervised clustering therefore amounts to dual supervision for clustering, i.e., both document supervision and feature supervision (Attenberg et al. 2010). Dual supervision takes place together and before the clustering algorithms begin. The clustering algorithms will use both labeled documents and features to guide the clustering process and produce clusters better matching user expectation.

Algorithm 5 Semi-supervised Clustering with Feature Supervision

Input: Set of data points $\mathcal{X}$
Output: K clusters $\{\mathcal{X}_l\}_{l=1}^{K}$
Method:

1: Perform *dual supervision*, i.e., *document supervision* and *feature supervision*
2: Obtain the accepted feature set $\mathcal{W}^{\mathcal{L}}$ and the document seed set $\mathcal{S}$ or must-link set $\mathcal{M}$ and cannot-link set $\mathcal{C}$
3: **if** Xing K-means **then**
4: Learn diagonal matrix $\mathcal{A}$ and set weights of accepted features to the maximum value in $\mathcal{A}$
5: Perform basic K-means clustering using the learned weights
6: **else**
7: Perform feature reweighting based on accepted feature set $\mathcal{W}^{\mathcal{L}}$.
8: Cluster the documents using semi-supervised clustering algorithm.
9: **end if**

5. EXPERIMENTAL RESULTS

In this section, we demonstrate the effectiveness of our proposed methods on several real-word data sets. Specifically, we study the performance of different weight values for feature reweighting, the size of the feature oracle vocabulary, the fraction of a document's content the user reads, and the noise level of the user (feature oracle). We enhance several semi-supervised clustering algorithms with feature supervision and compare algorithms with and without feature supervision.

5.1. Data Sets

We conducted our experiments on several real-word data sets of different sizes and also consisting of different types of text documents. We derive six data sets with different sizes and different separability from the 20-Newsgroup corpus[1] and three more data sets from webkb,[2] industry sector,[3] and reuters21578[4] separately. The descriptions and details

[1] http://people.csail.mit.edu/jrennie/20Newsgroups/

[2] http://www.cs.cmu.edu/~webkb

[3] http://www.cs.umass.edu/~mccallum/data.html

[4] http://kdd.ics.uci.edu

TABLE 1. Six Data Sets from 20-newsgroups, webkb, industry sectors, and reuters21578.

Data set	Description	Categories included	Categorized documents	Total documents
news-similar-3-100	The 20-Newsgroup data set consists of 20 different Usenet newsgroups, each of which has approximately 1,000 newsgroup messages.	3:comp.graphics, comp.os.ms, mswindows.misc, and comp.windows.x	100	300
news-diff-3-100		3:alt.atheism, rec.sport.baseball, and sci.space	100	300
news-related-3-100		3:talk.politics.misc, talk.politics.guns, and talk.politics.mideast	100	300
news-multi-7-100		7:alt.atheism, comp.sys.mac.hardware, misc.forsale, rec.sport.hockey, sci.crypt, talk.politics.guns, and soc.religion.christian	100	700
news-multi-10-100		10:alt.atheism,comp.sys.mac.hardware,misc.forsale, rec.autos,rec.sport.hockey,sci.crypt,sci.med, sci.electronics, sci.space, talk.politics.guns	100	1,000
webkb-sfcp-4-250	Web pages from different universities	4:student, faculty, course, and project	250	1,000
sector-multi-10-100	Web pages from different industrial sectors	10:basic.materials, capital.goods, consumer.cyclical, oil.and.gas.integrated, investment.services, biotechnology. and.drugs, hotels.and.motels, communications. equipment, railroad, and water.utilities	100 (railroad—95)	995
reuters-multi-10-100	News articles from reuters21578. We use the top 10 most frequent categories, the documents of which do not have multiple labels.	10:acq, coffee, crude, earn, gold, interest, money-fx, ship, sugar, and trade	100 (gold—90)	990

TABLE 2. All Variants of K-means with/without Document Supervision and Feature Supervision.

Algorithm	Document supervision (no/seed/constraint)	Feature supervision (yes/no)	Definition
Random K-means	No	No	The unsupervised K-means with random initialization of the centroids
Fes K-means	No	Yes	Performs feature supervision during defining document seeds or document constraints but does not use document seeds or document constraints for clustering
COP K-means	Constraint	No	Enforces document constraints during clustering process
COPFes K-means	Constraint	Yes	Enforces document constraints during clustering process and uses feature supervision during defining document constraints
Seeded K-means	Seed	No	Uses document seeds to initialize the centroids for K-means
Seeded Fes K-means	Seed	Yes	Uses document seeds to initialize the centroids for K-means and performs feature supervision during defining document seeds
Constrained K-means	Seed and constraint	No	Uses document seeds to initialize the centroids for K-means and constrain the clustering process using document constraints derived from document seeds
Constrained Fes K-means	Seed and constraint	Yes	Uses document seeds to initialize the centroids for K-means and constrain the clustering process using document constraints derived from document seeds. At the same time, feature supervision is performed during defining document seeds
Xing K-means	Constraint	No	Learns distance metric based on document constraints
Xing Fes K-means	Constraint	Yes	Learns distance metric and uses feature supervision

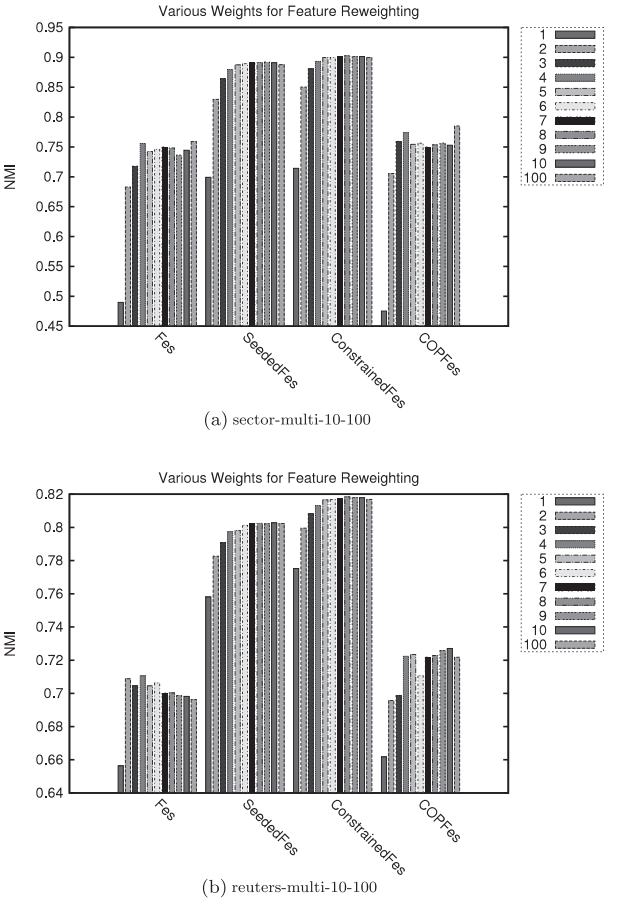
TABLE 3. Experiments We Ran and the Corresponding Setup Including Algorithms and Parameters.

Experiment	Algorithms	Weight g	Vocabulary f	Content fraction p_c	Noise fraction p_n	Number of seeds per cluster
Feature reweighting	Fes, seeded Fes, constrained Fes, and COPFes	1–10	30	1.0	0.0	10
Oracle capacity	All algorithms except Xing and Xing Fes	2	0–100	1.0	0.0	10
Content fraction	All algorithms except Xing and Xing Fes	2	30	0.0–1.0	0.0	10
Noise fraction	All algorithms except Xing and Xing Fes	2	30	1.0	0.0–1.0	10
Number of seeds	All algorithms except Xing and Xing Fes	2	30	1.0	0.0	0–50
Feature versus document supervision	All algorithms	2	30	1.0	0.0	10
Content and noise	All algorithms except Xing and Xing Fes	2	30	0.0–1.0	0.0–1.0	10

All algorithms are defined in Section 5.3.

TABLE 4. Experiments We Ran and the Corresponding Results.

Experiment	Results
Feature reweighting	Different data sets and algorithms achieved their best performance with different values of g. However, all weights used improve over their corresponding baselines ($g = 1$), namely, random K-means, seeded K-means, constrained K-means, and COP K-means.
Oracle capacity	The performance of the clustering algorithms stays relatively stable after the feature oracle vocabulary per cluster reaches a small size of 10–30. In practice, it means that the user does not have to know all the discriminative features, but only a few of the most discriminative ones.
Content fraction	The clustering performance only increases moderately with more than 10% of the content of a document being read. Therefore, the user does not need to read the whole content of a document for effective feature supervision.
Noise fraction	Even with some incorrect features being labeled as "accepted," the performance of semi-supervised clustering with feature supervision can still improve over the pure document supervision.
Number of seeds	Feature supervision with only a few defined documents as seeds or constraints can improve the clustering performance significantly compared with the pure document supervision method.
Featured versus document supervision	Random K-means with feature supervision only requires a few document constraints to be defined and features to be labeled to improve the clustering performance.
Content and noise	A noisy feature oracle still works very well even when only a small amount of the content of a document is read for defining seeds or document constraints. This observation allows human users to make mistakes in feature supervision while reading only part of a document and validates the practicality of our feature supervision model that feature supervision during document supervision can improve clustering performance.



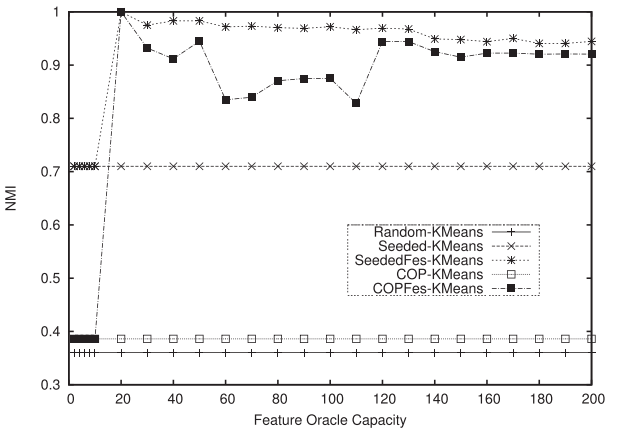
(a) sector-multi-10-100

(b) reuters-multi-10-100

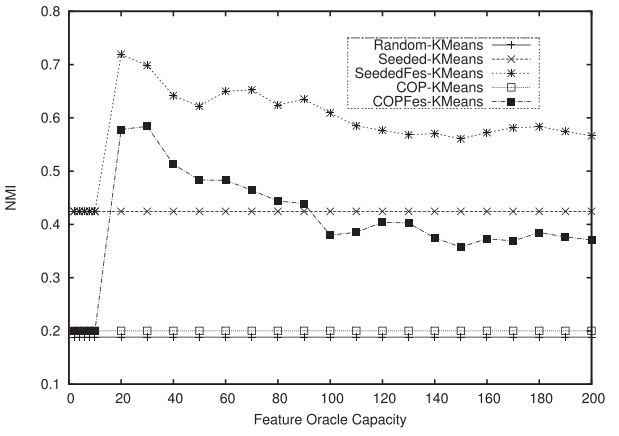
FIGURE 2. Feature reweighting with different weights: (a) sector-multi-10-100 and (b) reuters-multi-10-100. NMI, normalized mutual information.

TABLE 5. Feature Reweighting with Different Weights with Data Sets sector-multi-10-100 and reuters-multi-10-100.

Weight		1	2	3	4	5	6	7	8	9	10	100
Sector	Fes	0.32	0.40	0.42	0.41	0.40	0.40	0.39	0.39	0.40	0.40	0.39
	Seeded Fes	0.53	0.60	0.56	0.55	0.55	0.55	0.55	0.55	0.55	0.54	0.54
	Constrained Fes	0.56	0.59	0.60	0.59	0.58	0.58	0.58	0.58	0.57	0.57	0.57
	COPFes	0.35	0.43	0.44	0.43	0.41	0.41	0.42	0.42	0.41	0.40	0.41
Reuters	Fes	0.66	0.71	0.70	0.71	0.70	0.71	0.70	0.70	0.70	0.70	0.70
	Seeded Fes	0.76	0.78	0.79	0.80	0.80	0.80	0.80	0.80	0.80	0.80	0.80
	Constrained Fes	0.78	0.80	0.81	0.81	0.82	0.82	0.82	0.82	0.82	0.82	0.82
	COPFes	0.66	0.70	0.70	0.72	0.72	0.71	0.72	0.72	0.73	0.73	0.72



(a) news-diff-3-100



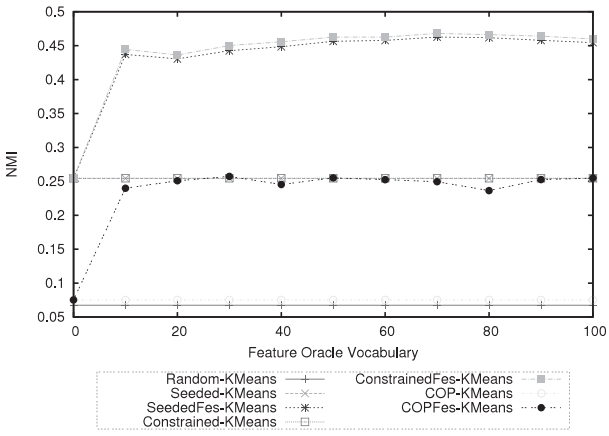
(b) news-related-3-100

FIGURE 3. Performance as a function of the size of feature vocabulary, i.e., feature oracle capacity. (a) news-diff-3-100 and (b) news-related-3-100. NMI, normalized mutual information.

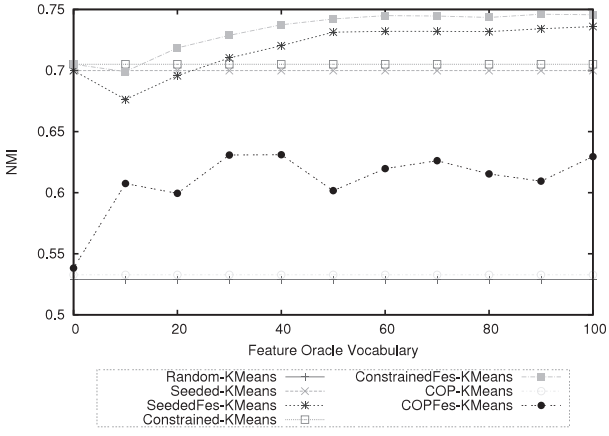
of the data sets are summarized in Table 1. Particularly, data set *news-diff-3* covers topics from three quite different newsgroups (alt.atheism, rec.sport.baseball, and sci.space). Data set *news-related-3* contains three related newsgroups (talk.politics.misc, talk.politics.guns, and talk.politics.mideast). Data set *news-similar-3* consists of messages from three similar newsgroups (comp.graphics, comp.os.ms-windows, and comp.windows.x). Because *news-similar-3* has a significant overlap between groups, it is the most difficult one to be clustered. Those three data sets are created for the purpose of studying the effect of data separability of the algorithms. Other data sets are generated for the purpose of studying the effect of data set size on the performance of the algorithms.

We preprocessed each document by tokenizing the text into bags-of-words.[5] Then, we removed the stop words and stemmed all the remaining words. Next, we selected the top 2000 words using mutual information between words and documents (Dhillon, Mallela, and Modha 2003). Finally, a feature vector for each document is constructed with TFIDF weighting and then normalized.

[5] A word is defined as a sequence of alphabetic characters delimited by nonalphabetic characters.



(a) news-similar-3-100



(b) news-multi-7-100

FIGURE 4. Performance as a function of the size of feature vocabulary, i.e., feature oracle capacity. (a) news-similar-3-100 and (b) news-multi-7-100. NMI, normalized mutual information.

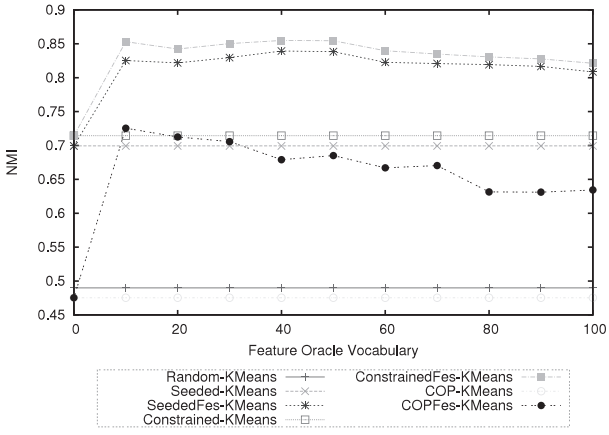
5.2. Evaluation Measures

Normalized mutual information (NMI) (Dom 2001) measures the shared information between the cluster assignments S and class labels L of documents. It is defined as

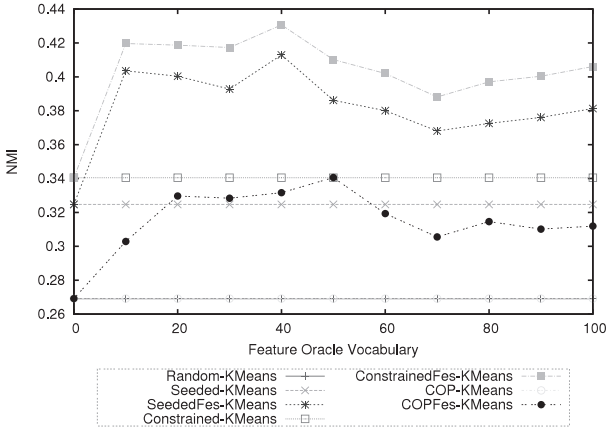
$$NMI(S, L) = \frac{I(S, L)}{(H(S) + H(L))/2} \tag{9}$$

where $I(S, L)$, $H(S)$, and $H(L)$ denote the mutual information between S and L, the entropy of S, and the entropy of L, respectively. Assuming there are K classes, K clusters, and N documents; $n(l_i)$ denotes the number of documents in class l_i; $n(s_j)$ denotes the number of documents in cluster s_j; and $n(l_i, s_j)$ denotes the number of documents in both class l_i and cluster s_j, we define

$$H(L) = -\sum_{i=1}^{K} P(l_i) \log_2 P(l_i) \tag{10}$$



(a) news-multi-10-100



(b) webkb-sfcp-4-250

FIGURE 5. Performance as a function of the size of feature vocabulary, i.e., feature oracle capacity. (a) news-multi-10-100 and (b) webkb-sfcp-4-250. NMI, normalized mutual information.

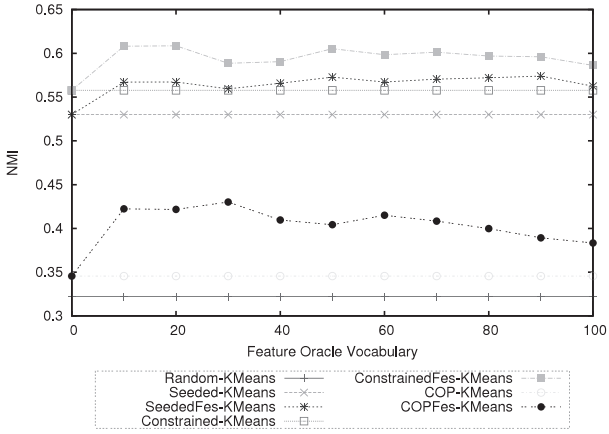
$$H(S) = -\sum_{j=1}^{K} P(s_j) \log_2 P(s_j) \tag{11}$$

$$I(S, L) = -\sum_{i=1}^{K} \sum_{j=1}^{K} P(l_i, s_j) \log_2 \frac{P(l_i, s_j)}{P(l_i) P(s_j)} \tag{12}$$

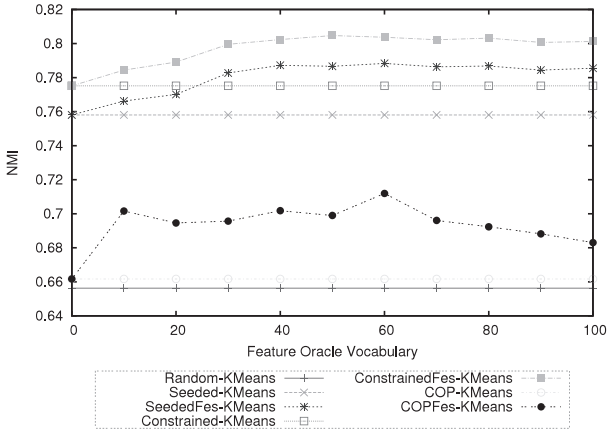
where $P(l_i) = n(l_i)/N$, $P(s_j) = n(s_j)/N$ and $P(l_i, s_j) = n(l_i, s_j)/N$. The NMI values are in the interval $[0, 1]$.

5.3. Clustering Algorithms

In this article, we have several variants of K-means with document supervision and/or feature supervision for our experiments. The algorithms and the corresponding supervision methods are summarized in Table 2.



(a) sector-multi-10-100



(b) reuters-multi-10-100

FIGURE 6. Performance as a function of the size of feature vocabulary, i.e., feature oracle capacity. (a) sector-multi-10-100 and (b) reuters-multi-10-100. NMI, normalized mutual information.

5.4. Parameters for the Experiments

In this subsection, we describe the free parameters used in the experiments. The actual values of the parameters are given in the analysis of the results in Section 5.5.

Feature reweighting g**:** Feature reweighting and weight g are defined in Section 4.4. Different values of g might lead to different clusterings.

Feature oracle capacity f**:** This is the number of features the feature oracle can recognize and label as "accept," namely, the size of the feature oracle vocabulary. Because we do not know the best value of f for clustering, we conducted experiments with different values of f. The f features that the user labels as "accepted" are included in the feature oracle vocabulary. The general hypothesis is values of s that are neither too large nor too small can produce good clusters. Many noise features are included when s is too large (more than half of all extracted features), while many useful features for clustering are excluded when s is too small (less than 10).

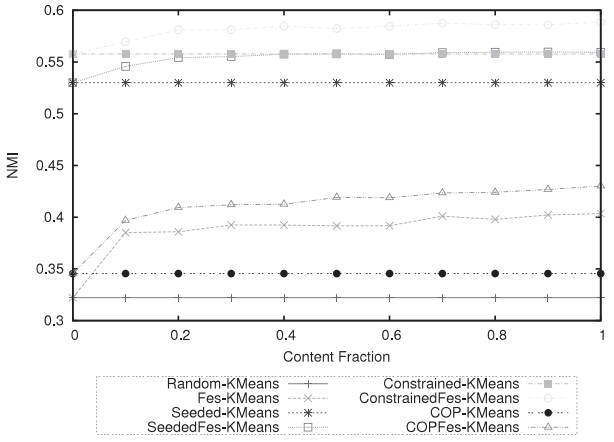
TABLE 6. Performance as a Function of the Size of Feature Vocabulary, i.e., Feature Oracle Capacity.

Capacity		0	20	40	60	80	100
news-similar-3-100	Seeded Fes	0.44	0.43	0.45	0.46	0.46	0.45
	Constrained Fes	0.44	0.44	0.46	0.46	0.47	0.46
	COPFes	0.24	0.25	0.25	0.25	0.24	0.25
news-multi-7-100	Seeded Fes	0.70	0.70	0.72	0.73	0.73	0.74
	Constrained Fes	0.71	0.72	0.74	0.74	0.74	0.75
	COPFes	0.54	0.60	0.63	0.62	0.62	0.63
news-multi-10-100	Seeded Fes	0.70	0.82	0.84	0.82	0.82	0.81
	Constrained Fes	0.71	0.84	0.85	0.84	0.83	0.82
	COPFes	0.48	0.71	0.68	0.67	0.63	0.63
webkb-sfcp-4-250	Seeded Fes	0.32	0.40	0.41	0.38	0.37	0.38
	Constrained Fes	0.34	0.42	0.43	0.40	0.40	0.41
	COPFes	0.27	0.33	0.33	0.32	0.31	0.31
sector-multi-10-100	Seeded Fes	0.53	0.57	0.57	0.57	0.57	0.56
	Constrained Fes	0.56	0.61	0.59	0.60	0.60	0.59
	COPFes	0.35	0.42	0.41	0.42	0.40	0.38
reuters-multi-10-100	Seeded Fes	0.76	0.77	0.79	0.79	0.79	0.79
	Constrained Fes	0.78	0.79	0.80	0.80	0.80	0.80
	COPFes	0.66	0.69	0.70	0.71	0.69	0.68

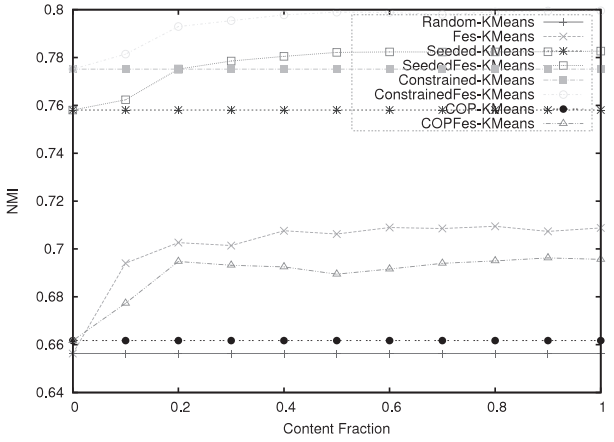
Content fraction p_c: Because the user does not have to read the whole content of a document to label it, we assume that the user reads a fraction p_c of its content starting from the beginning of the document. The general hypothesis is that the more content the user reads, the more features the user will label and the better the performance will be, provided that the the user can label the features correctly. However, if the user is not confident with feature labeling, reading more content might not help or even harm the clustering performance.

Noisy feature fraction p_n: Because the user can make mistakes by accepting poor features for clustering, we constructed feature oracles with various fractions of noisy features (Algorithm 4). The general hypothesis is that the more noisy the feature oracle, the worse the clustering performance.

Number of seeds or constraints per cluster: We used different numbers of cluster seeds and constraints for the semi-supervised clustering algorithms. The cluster seeds for seeded K-means and constrained K-means are randomly sampled and defined from the documents belonging to the corresponding class. Because we compare the COP K-means, seeded K-means, and constrained K-means, *the constraints used for COP K-means* are constructed from the cluster seeds by establishing must-link constraints between the seeds with the same cluster labels and by establishing cannot-link constraints between the seeds with different cluster labels. *The constraints for Xing K-means* are randomly sampled.



(a) sector-multi-10-100



(b) reuters-multi-10-100

FIGURE 7. Enhanced with feature supervision with varying content being read. (a) sector-multi-10-100 and (b) reuters-multi-10-100. NMI, normalized mutual information.

5.5. Analysis of Results

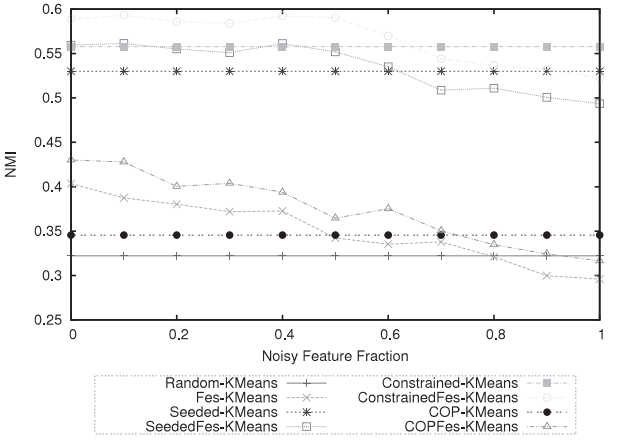
Except when explicitly stated, we assume the whole content is read to define a document seed or a pairwise constraint and a noise-free feature oracle is employed to label the words in the documents. In addition, we set the number of seeds for each cluster to 10 and the feature capacity per cluster f to 30 if not explicitly described.

For the clarity of the article, we are not able to present all the experimental results for all combinations of data sets and algorithms. Therefore, we mainly use data set sector-multi-10-100 and seeded K-means (if not explicitly stated) to illustrate our points. The results for all other data sets have a similar pattern to those presented here. However, we include the results of all data sets for discussion of the feature oracle capacity and number of seeds for completeness. The experiments we ran and the setup are summarized in Table 3. The results are summarized in Table 4.

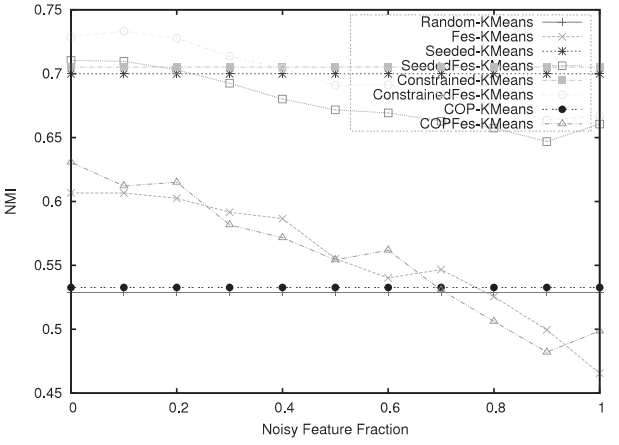
5.5.1. Feature Reweighting g. Different weight values, g (refer to Section 4.4 for details), might lead to different clustering results. We conducted experiments with different

TABLE 7. Enhanced with Feature Supervision with Varying Content Being Read.

		Content fraction					
		0.0	0.2	0.4	0.6	0.8	1.0
sector-multi-10-100	Fes	0.32	0.39	0.39	0.39	0.40	0.40
	Seeded Fes	0.53	0.55	0.56	0.56	0.56	0.56
	Constrained Fes	0.56	0.58	0.58	0.58	0.59	0.59
	COPFes	0.35	0.40	0.41	0.42	0.42	0.43
reuters-multi-10-100	Fes	0.66	0.70	0.71	0.71	0.71	0.71
	Seeded Fes	0.76	0.78	0.78	0.78	0.78	0.78
	Constrained Fes	0.78	0.79	0.80	0.80	0.80	0.80
	COPFes	0.66	0.69	0.69	0.69	0.70	0.70



(a) sector-multi-10-100

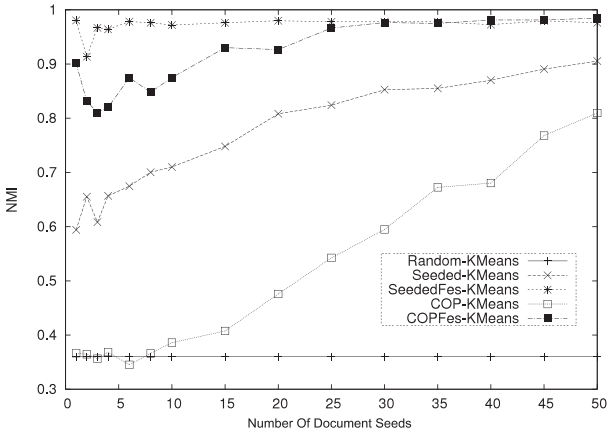


(b) news-multi-7-100

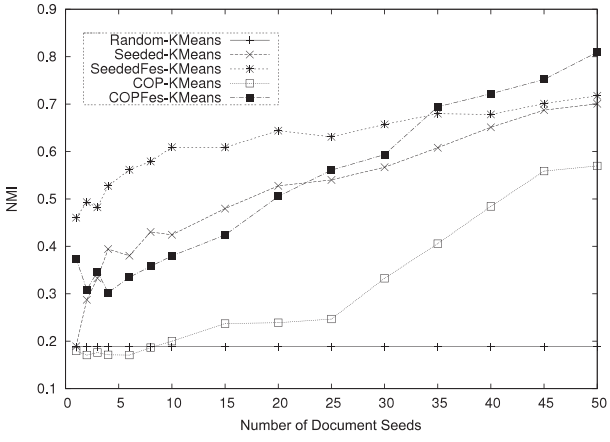
FIGURE 8. Enhanced with feature supervision with varying noise feature fractions. (a) sector-multi-10-100 and (b) news-multi-7-100. NMI, normalized mutual information.

TABLE 8. Enhanced with Feature Supervision with Varying Noise Feature Fractions.

		Noisy fraction					
		0.0	0.2	0.4	0.6	0.8	1.0
sector-multi-10-100	Fes	0.40	0.38	0.37	0.34	0.32	0.30
	Seeded Fes	0.56	0.56	0.56	0.54	0.51	0.49
	Constrained Fes	0.59	0.59	0.59	0.57	0.54	0.52
	COPFes	0.43	0.40	0.39	0.38	0.33	0.32
news-multi-7-100	Fes	0.61	0.60	0.59	0.54	0.53	0.47
	Seeded Fes	0.71	0.70	0.68	0.67	0.66	0.66
	Constrained Fes	0.73	0.73	0.70	0.69	0.68	0.67
	COPFes	0.63	0.62	0.57	0.56	0.51	0.50

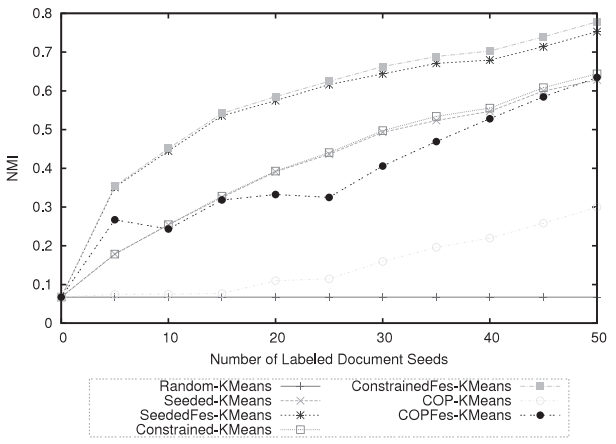


(a) news-diff-3-100

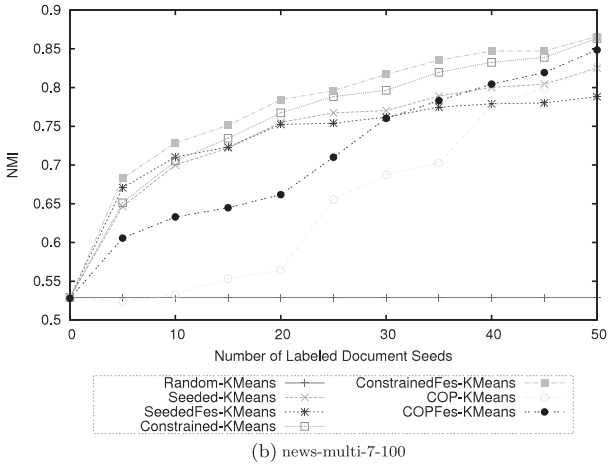


(b) news-related-3-100

FIGURE 9. Different numbers of document seeds (constraints for COP K-means and COPFes K-means are generated from document seeds; refer to Section 5.4 for details). (a) news-diff-3-100 and (b) news-related-3-100. NMI, normalized mutual information.



(a) news-similar-3-100



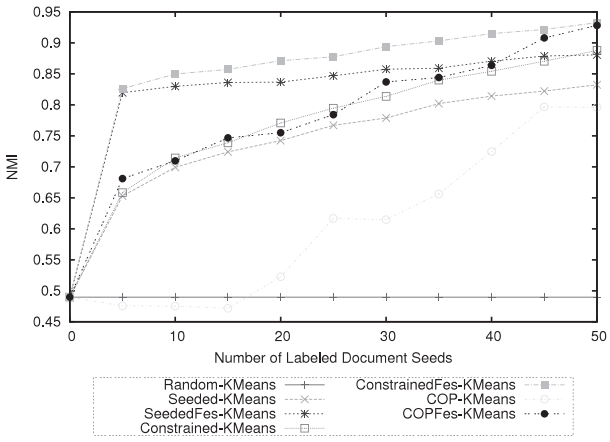
(b) news-multi-7-100

FIGURE 10. Different numbers of document seeds (constraints for COP K-means and COPFes K-means are generated from document seeds; refer to Section 5.4 for details). (a) news-similar-3-100 and (b) news-multi-7-100. NMI, normalized mutual information.

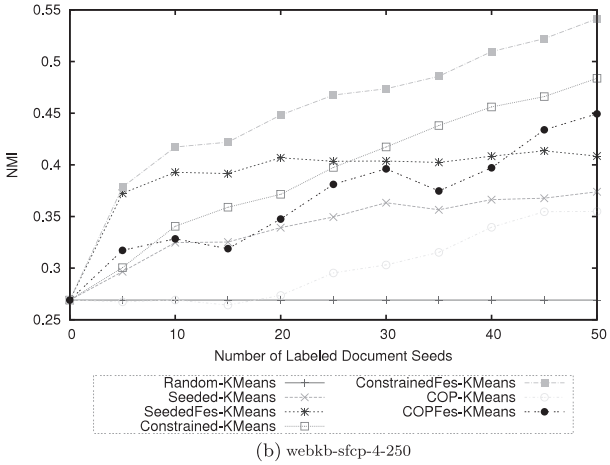
values of g to show the robustness of our algorithms. Results show that different data sets and algorithms achieved their best performance with different values of g (Figure 2 and Table 5). However, all weights used improve over their corresponding baselines ($g = 1$), namely, random K-means, seeded K-means, constrained K-means, and COP K-means. Because of the limit of space, we select $g = 2$ to report the results on the following experiments. Weight 2 is selected because it is seldom the weight to achieve the best performance for various algorithms on all data sets. Namely, we give the benefit to the baseline algorithms.

5.5.2. Feature Oracle Capacity f. Assuming the whole content of a document seed (or two documents in a pairwise constraint) is read and a noise-free feature oracle, semi-supervised clustering with feature supervision shows significantly[6] improved performance

[6] Two-tailed paired t-test with $p = 0.05$. Also applies to other significance statements.



(a) news-multi-10-100



(b) webkb-sfcp-4-250

FIGURE 11. Different numbers of document seeds (constraints for COP K-means and COPFes K-means are generated from document seeds; refer to Section 5.4 for details). (a) news-multi-10-100 and (b) webkb-sfcp-4-250. NMI, normalized mutual information.

over the method without feature supervision (Figures 3–6 and Table 6). With feature supervision, constrained K-means and seeded K-means still work much better than COP K-means. It is noticeable that the performance of the clustering algorithms stays relatively stable after the feature oracle vocabulary per cluster reaches a small size of 10–30. In practice, it means that the user does not have to know all the discriminative features, but only a few of the most discriminative ones. As f grows, clustering performances may decrease, e.g., Figure 5(b). Because the algorithm used to construct the feature oracle is not perfect, it is unavoidable to include some features that are not discriminating for clustering in the feature oracle vocabulary as f grows. We conjecture that clustering performance declines because of the presence of such features introduced by the construction algorithm. The behavior of a noisy feature oracle with explicitly injected poor features is explored later.

5.5.3. Content Fraction p_c. Assuming a noise-free feature oracle, the clustering performance with feature supervision is improved with more content of a defined document being read (Figure 7 and Table 7). At the same time, regardless of the fraction of the content

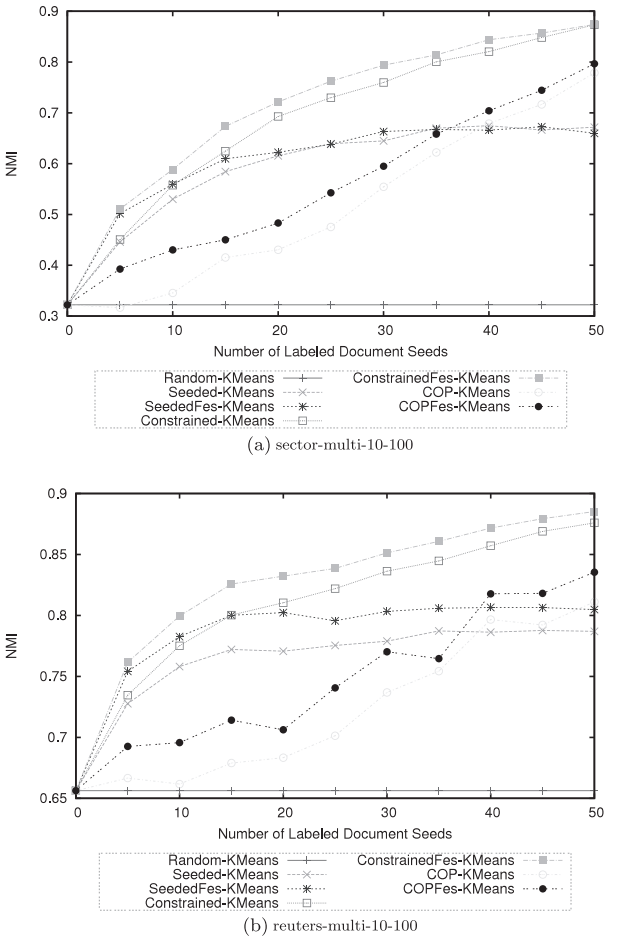


FIGURE 12. Different numbers of document seeds (constraints for COP K-means and COPFes K-means are generated from document seeds; refer to Section 5.4 for details). (a) sector-multi-10-100 and (b) reuters-multi-10-100. NMI, normalized mutual information.

read (at least 10% in our experiments), the performance of semi-supervised clustering with feature supervision is much better than that of the method with only defined constraints. In fact, the clustering performance only increases moderately with more than 10% of the content of a document being read. Therefore, the user does not need to read the whole content of a document for effective feature supervision, just as the user does not have to read the whole content to define a document.

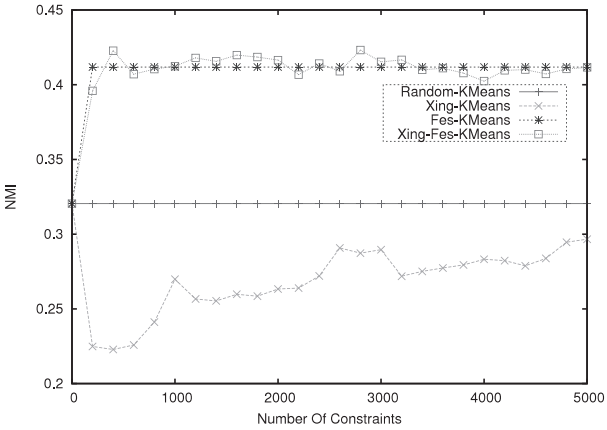
5.5.4. Noisy Feature Fraction p_n. Assuming the whole content of a defined document being read, we study the behavior of the noisy feature oracle, which can make mistakes in labeling features. Through the experiments, we find that the clustering performance decreases as more noisy features are introduced by the feature oracle, namely, the more mistakes the feature oracle makes, the worse the performance is (Figure 8 and Table 8). However, even with some incorrect features being labeled as "accepted," the performance of semi-supervised clustering with feature supervision can still improve over the pure document supervision. In fact, it is demonstrated that our algorithms have high tolerance of mistakes in labeling features (Figure 8). It may be because very few accepted

TABLE 9. Different Numbers of Document Seeds (Constraints for COP K-means and COPFes K-means Are Generated from Document Seeds; Refer to Section 5.4 for Details).

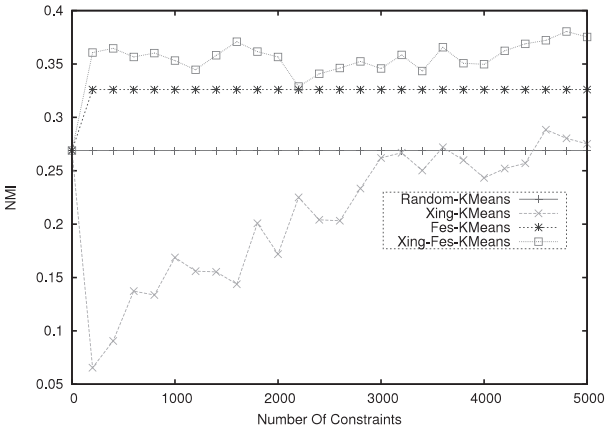
		No. of seeds					
		0	10	20	30	40	50
news-similar-3-100	Seeded	0.07	0.32	0.39	0.49	0.55	0.63
	Seeded Fes	0.07	0.54	0.57	0.64	0.68	0.75
	Constrained	0.07	0.33	0.39	0.50	0.56	0.64
	Constrained Fes	0.07	0.54	0.59	0.66	0.70	0.78
	COP	0.07	0.08	0.11	0.16	0.22	0.30
	COPFes	0.07	0.24	0.33	0.41	0.53	0.63
news-multi-7-100	Seeded	0.53	0.70	0.76	0.77	0.80	0.83
	Seeded Fes	0.53	0.71	0.75	0.76	0.78	0.79
	Constrained	0.53	0.71	0.77	0.80	0.83	0.86
	Constrained Fes	0.53	0.73	0.78	0.82	0.85	0.87
	COP	0.53	0.53	0.56	0.69	0.78	0.83
	COPFes	0.53	0.63	0.66	0.76	0.80	0.85
news-multi-10-100	Seeded	0.49	0.70	0.74	0.78	0.81	0.83
	Seeded Fes	0.49	0.83	0.84	0.86	0.87	0.88
	Constrained	0.49	0.71	0.77	0.81	0.85	0.89
	Constrained Fes	0.49	0.85	0.87	0.89	0.91	0.93
	COP	0.49	0.48	0.52	0.61	0.72	0.80
	COPFes	0.49	0.71	0.76	0.84	0.86	0.93
webkb-sfcp-4-250	Seeded	0.27	0.32	0.34	0.36	0.37	0.37
	Seeded Fes	0.27	0.39	0.41	0.40	0.41	0.41
	Constrained	0.27	0.34	0.37	0.42	0.46	0.48
	Constrained Fes	0.27	0.42	0.45	0.47	0.51	0.54
	COP	0.27	0.27	0.27	0.30	0.34	0.35
	COPFes	0.27	0.33	0.35	0.40	0.40	0.45
sector-multi-10-100	Seeded	0.32	0.53	0.62	0.64	0.67	0.67
	Seeded Fes	0.32	0.56	0.62	0.66	0.67	0.66
	Constrained	0.32	0.56	0.72	0.76	0.82	0.87
	Constrained Fes	0.32	0.59	0.43	0.79	0.84	0.87
	COP	0.32	0.35	0.43	0.55	0.68	0.78
	COPFes	0.32	0.43	0.48	0.59	0.70	0.80
reuters-multi-10-100	Seeded	0.66	0.76	0.77	0.78	0.79	0.79
	Seeded Fes	0.66	0.78	0.80	0.80	0.81	0.80
	Constrained	0.66	0.78	0.81	0.84	0.86	0.88
	Constrained Fes	0.66	0.80	0.83	0.85	0.87	0.89
	COP	0.66	0.66	0.68	0.74	0.80	0.81
	COPFes	0.66	0.70	0.71	0.77	0.82	0.84

TABLE 10. Metric Learning Method and Feature Supervision Method.

		No. of constraints					
		0	1,000	2,000	3,000	4,000	5,000
sector-multi-10-100	Xing	0.32	0.27	0.26	0.29	0.28	0.30
	Fes	0.32	0.41	0.41	0.41	0.41	0.41
	Xing Fes	0.32	0.41	0.42	0.42	0.40	0.41
webkb-multi-4-250	Xing	0.27	0.17	0.17	0.26	0.24	0.28
	Fes	0.27	0.33	0.33	0.33	0.33	0.33
	Xing Fes	0.27	0.35	0.36	0.35	0.35	0.38

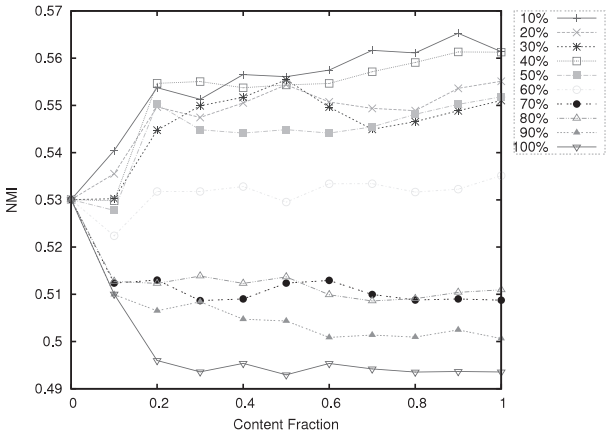


(a) sector-multi-10-100

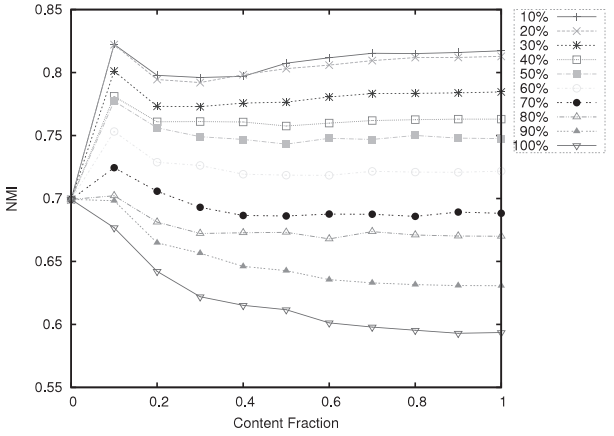


(b) webkb-sfcp-4-250

FIGURE 13. Metric learning method and feature supervision method. (a) sector-multi-10-100 and (b) webkb-sfcp-4-250. NMI, normalized mutual information.



(a) sector-multi-10-100

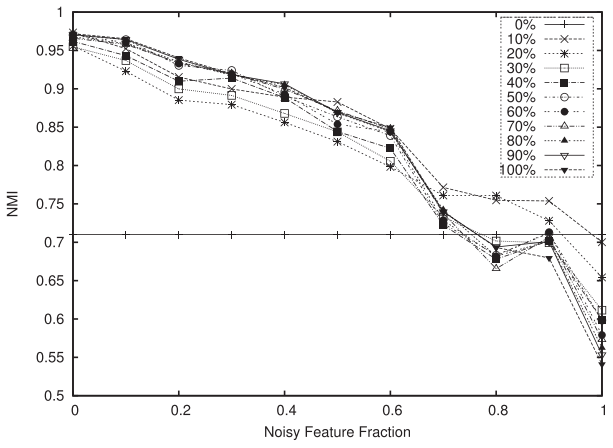


(b) news-multi-10-100

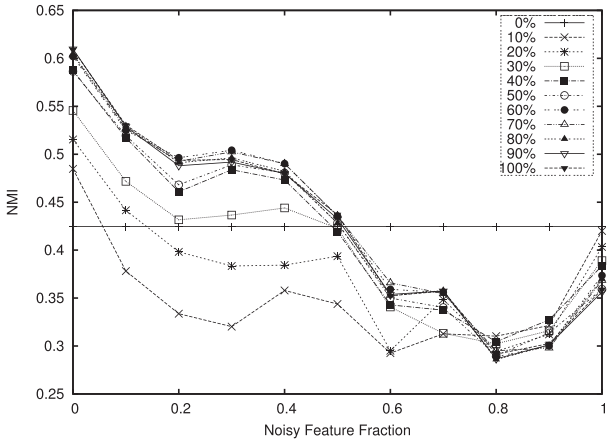
FIGURE 14. Seeded Fes K-means with varying content being read for feature oracle with different noisy feature levels. Each curve represents a feature oracle with the corresponding level of noisy features. (a) sector-multi- 10-100 and (b) news-multi-10-100. NMI, normalized mutual information.

features that are highly discriminative dominate the clustering despite the presence of many nondiscriminative features.

5.5.5. Number of Seeds or Constraints per Cluster. Feature supervision with only a few documents defined can improve the clustering performance significantly compared with the pure document supervision method (Figures 9–12 and Table 9). To achieve the same performance without feature supervision, many more documents have to be defined. For example, 20 documents per cluster have to be defined as seeds to achieve the same performance as 15 documents per cluster defined with feature supervision (Figure 12(a)). With more documents defined, feature supervision becomes less important than when there are only few defined documents. This implies that feature supervision can help us save user effort from defining unnecessary documents. Because the user labels features while defining documents, feature supervision in our proposed methods does not have to involve much extra effort.



(a) news-diff-3-100



(b) news-related-3-100

FIGURE 15. Seeded Fes K-means with feature oracle with different noisy feature levels with varying content being read. Each curve represents a certain percentage of content being read. (a) news-diff-3-100 and (b) news-related-3-100. NMI, normalized mutual information.

5.5.6. Feature Supervision versus Document Supervision. Besides the semi-supervised clustering with/without feature supervision, we also ran the random K-means only with the accepted features, i.e., Fes K-means, the algorithm used by Hu et al. (2011) to incorporate accepted features. Random K-means with feature supervision works better than COP K-means and comparatively with COPFes K-means (Figure 8 and Table 10). Although Fes K-means works worse than seeded K-means and constrained K-means, semi-supervised clustering with feature supervision always works better than without feature supervision on all data sets. The distance metric learning method based on defined document constraints works much worse than random K-means even when quite a large number of constraints are given (Figure 13). Our explanation is that the high-dimensional and sparse document vectors require too many document constraints to learn a correct distance metric. With only a few document constraints, some unimportant features are unavoidably overweighted. However, random K-means with feature supervision only requires a few constraints and features to be defined to improve the clustering performance. Note that Xing

TABLE 11. Seeded Fes K-means with Varying Content Being Read for Feature Oracle with Different Noisy Feature Levels.

		Content fraction					
		0.0	0.2	0.4	0.6	0.8	1.0
sector-multi-10-100	10%	0.53	0.55	0.56	0.56	0.56	0.56
	20%	0.53	0.55	0.55	0.55	0.55	0.56
	30%	0.53	0.54	0.55	0.55	0.55	0.55
	40%	0.53	0.55	0.55	0.55	0.56	0.56
	50%	0.53	0.55	0.54	0.54	0.55	0.55
	60%	0.53	0.53	0.53	0.53	0.53	0.54
	70%	0.53	0.51	0.51	0.51	0.51	0.51
	80%	0.53	0.51	0.51	0.51	0.51	0.51
	90%	0.53	0.51	0.50	0.50	0.50	0.50
	100%	0.53	0.50	0.50	0.50	0.49	0.49
new-multi-10-100	10%	0.70	0.80	0.80	0.81	0.82	0.82
	20%	0.70	0.79	0.80	0.81	0.81	0.81
	30%	0.70	0.77	0.78	0.78	0.78	0.78
	40%	0.70	0.76	0.76	0.76	0.76	0.76
	50%	0.70	0.76	0.75	0.75	0.75	0.75
	60%	0.70	0.73	0.72	0.72	0.72	0.72
	70%	0.70	0.71	0.69	0.69	0.69	0.69
	80%	0.70	0.68	0.67	0.67	0.67	0.67
	90%	0.70	0.66	0.65	0.64	0.63	0.63
	100%	0.70	0.64	0.62	0.60	0.60	0.59

Each row represents a feature oracle with the corresponding level of noisy features.

Fes K-means can still improve the clustering performance further compared with Fes K-means. However, the Euclidean distance metric learning algorithm is quite computationally expensive (hours for metric learning versus seconds for feature reweighting for labeled features) even when a diagonal matrix is assumed because of the high-dimensional vector representation of documents.

5.5.7. Noisy Feature Oracle and Content Fraction. Instead of assuming a noise-free feature oracle and that the user reads the whole content of a document to define it, we explore the behavior of the noisy feature oracle while only part of content is read to define a document. In Figure 14, each curve represents the clustering performance of a noisy feature oracle with different noise levels when different fractions of content are read. It is verified that the clustering performance improves as the user reads more content of a defined document and when the feature oracle is less noisy (Figures 14 and 15 and Tables 11 and 12). More importantly, those figures demonstrate that a noisy feature oracle still works very well even when only a small amount of content of a document is read. This observation allows human users to make mistakes in feature supervision while reading only part of the content for defining a document and validates the practicality of our feature supervision model that feature supervision during document supervision can improve clustering performance. However, for a very noisy feature oracle, such as one with 80% noisy features (Figure 14),

TABLE 12. Seeded Fes K-means with Feature Oracle with Different Noisy Feature Levels with Varying Content Being Read.

		Content fraction					
		0.0	0.2	0.4	0.6	0.8	1.0
sector-multi-10-100	10%	0.56	0.54	0.53	0.52	0.51	0.51
	20%	0.56	0.55	0.55	0.53	0.51	0.50
	30%	0.56	0.55	0.56	0.53	0.51	0.49
	40%	0.56	0.55	0.55	0.53	0.51	0.50
	50%	0.56	0.55	0.55	0.53	0.51	0.49
	60%	0.56	0.55	0.55	0.53	0.51	0.50
	70%	0.56	0.55	0.56	0.53	0.51	0.49
	80%	0.56	0.55	0.56	0.53	0.51	0.49
	90%	0.56	0.55	0.56	0.53	0.51	0.49
	100%	0.56	0.56	0.56	0.54	0.51	0.49
reuters-multi-10-100	10%	0.76	0.74	0.73	0.72	0.71	0.75
	20%	0.78	0.75	0.72	0.71	0.70	0.75
	30%	0.78	0.75	0.73	0.72	0.69	0.75
	40%	0.78	0.75	0.73	0.72	0.70	0.75
	50%	0.78	0.75	0.73	0.72	0.69	0.75
	60%	0.78	0.75	0.73	0.72	0.70	0.75
	70%	0.78	0.77	0.73	0.72	0.70	0.74
	80%	0.78	0.76	0.73	0.72	0.70	0.74
	90%	0.78	0.76	0.73	0.72	0.70	0.74
	100%	0.78	0.76	0.73	0.72	0.70	0.74

Each row represents a certain percentage of content being read.

the clustering performance decreases when more content of a document is read, because the more content is read, the more noisy features are introduced. Because of the limit of space, only the results for seeded K-means are presented. The results for Fes K-means, COPFes K-means, and constrained Fes K-means have similar patterns.

6. CONCLUSIONS AND FUTURE WORK

In this article, we enhance the traditional semi-supervised document clustering with feature supervision, which asks the user to label features by indicating whether they discriminate among clusters. We make the assumption that the user can label features while he is defining a document so that the discriminating features are obtained without too much extra work. The labeled features are incorporated into semi-supervised clustering by feature reweighting, which explicitly gives more weight to the features that, according to the user, discriminate among clusters. We explore this enhancement by employing different types of semi-supervised clustering algorithms. Experimental results demonstrate that all types of semi-supervised clustering algorithms enhanced with feature supervision improved clustering performance significantly. Specifically, the distance metric learned using feature supervision on top of document constraints works significantly better than the one learned

based only on document constraints. We also find that feature supervision improves clustering performance even when only a small amount of content of the defined documents is read and some mistakes are made in labeling features.

In this article, we discuss how to augment semi-supervised clustering based on document-level supervision with feature-level supervision. We experimented with three different types of traditional semi-supervised clustering algorithms: (1) constraint-based methods, (2) seeding methods, and (3) distance-based methods. To complete this work, we would experiment with one hybrid method (Basu et al. 2004).

By applying distance metric learning to text clustering, we found that too many constraints are needed before effective weights are learned. Therefore, we conjecture that it is not suitable to use metric learning based on defined document constraints when there are not enough constraints for the high-dimensional space vectors. We plan to experiment with more algorithms involving metric learning based on document constraints only (Bar-Hillel et al. 2003) or both constraints and intermediate clusters (Basu et al. 2004). In this article, we assume the feature supervision takes place during document supervision. We can separate those two processes and interleave active document selection (Huang and Lam 2009) and active feature selection (Hu et al. 2011).

ACKNOWLEDGMENTS

We would like to thank anonymous reviewers for their insightful comments. This research was supported in part by the the Natural Sciences and Engineering Research Council Business Intelligence Network and by the Mathematics of Information Technology and Complex Systems Network of Centers of Excellence.

REFERENCES

ATTENBERG, J., P. MELVILLE, and F. PROVOST. 2010. A unified approach to active dual supervision for labeling features and examples. *In* ECML PKDD 2010 Part I, LNAI 6321, Barcelona, Spain, pp. 40–55.

BAR-HILLEL, A., T. HERTZ, N. SHENTAL, and D. WEINSHALL. 2003. Learning distance functions using equivalence relations. *In* International Conference on Machine Learning, Vol. 20, Washington, DC, p. 11.

BASU, S., A. BANERJEE, and R. MOONEY. 2002. Semi-supervised clustering by seeding. *In* International Conference on Machine Learning, Sydney, Australia, pp. 19–26.

BASU, S., M. BILENKO, and R. J. MOONEY. 2004. A probabilistic framework for semi-supervised clustering. *In* Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, WA, pp. 59–68.

CHENG, H., K. A. HUA, and K. VU. 2008. Constrained locally weighted clustering. Proceedings of VLDB'08, **1**(1): 90–101.

DHILLON, I. S., S. MALLELA, and D. S. MODHA. 2003. Information-theoretic co-clustering. *In* Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, pp. 89–98.

DOM, B. E. 2001. An information-theoretic external cluster-validity measure. Technical Report RJ 10219, IBM Research Division.

DRUCK, G., G. MANN, and A. MCCALLUM. 2008. Learning from labeled features using generalized expectation criteria. *In* Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Singapore, pp. 595–602.

HU, Y., E. E. MILIOS, and J. BLUSTEIN. 2011. Interactive feature selection for document clustering. *In* Proceedings of the 26th Symposium on Applied Computing, on Track "Information Access and Retrieval," ACM Special Interest Group on Applied Computing, Taichung, Taiwan, pp. 1148–1155.

HU, Y., E. E. MILIOS, and J. BLUSTEIN. 2012. Enhancing semi-supervised document clustering with feature supervision. *In* Proceedings of the 27th ACM Symposium Applied Computing, On Track "Information Access and Retrieval," Riva del Garda, Italy, pp. 950–957.

HUANG, R., and W. LAM. 2009. An active learning framework for semi-supervised document clustering with language modeling. Data & Knowledge Engineering, **68**(1): 49–67.

HUANG, Y., and T. M. MITCHELL. 2006. Text clustering with extended user feedback. *In* Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Seattle, WA, p. 420.

JI, X., and W. XU. 2006. Document clustering with prior knowledge. *In* Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Seattle, WA, p. 412.

LAMANTIA, J. 2007. Text clouds: a new form of tag cloud? Available at: http://www.joelamantia.com/tag-clouds/text-clouds-a-new-form-of-tag-cloud. Accessed April 12, 2012.

LIU, B., X. LI, W. S. LEE, and P. S. YU. 2004. Text classification by labeling words. *In* Proceedings of the National Conference on Artificial Intelligence, San Jose, CA, pp. 425–430.

RAGHAVAN, H., O. MADANI, and R. JONES. 2005. Interactive feature selection. *In* Proceedings of IJCAI 05: The 19th International Joint Conference on Artificial Intelligence, Edinburgh, UK, pp. 841–846.

TANG, W., H. XIONG, S. ZHONG, and J. WU. 2007. Enhancing semi-supervised clustering: a feature projection perspective. *In* Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Jose, CA, pp. 707–716.

WAGSTAFF, K., C. CARDIE, S. ROGERS, and S. SCHRÖDL. 2001. Constrained k-means clustering with background knowledge. *In* Proceedings of the Eighteenth International Conference on Machine Learning, Williamstown, MA, pp. 577–584.

XING, E. P., A. Y. NG, M. I. JORDAN, and S. RUSSELL. 2003. Distance metric learning with application to clustering with side-information. *In* Advances in Neural Information Processing Systems, Vancouver, BC, pp. 521–528.

YOSHIDA, T. 2012. A graph-based approach for semisupervised clustering. Computational Intelligence, **30**(2): 263–284.